

Article

# Tip-Over Detection and Avoidance Algorithms as Stabilization Strategy for Small-Footprint and Lightweight Mobile Manipulators

Aishe Toledo Fuentes <sup>1,2</sup>, Franziska Kempf <sup>1</sup>, Martin Kipfmüller <sup>1,\*</sup>, Tobias Bergmann <sup>1</sup> and Miguel J. Prieto <sup>2</sup> 

<sup>1</sup> Department of Mechanical Engineering and Mechatronics, University of Applied Science Karlsruhe, Moltkestraße 30, 76133 Karlsruhe, Germany

<sup>2</sup> Electrical Engineering Department, University of Oviedo, Gijón Campus EDO 3, 33204 Gijón, Spain

\* Correspondence: martin.kipfmuller@h-ka.de; Tel.: +49-721-925-1905

**Abstract:** The risk of tip-over is a common problem in agile, lightweight mobile manipulators. An easy-to-implement and reliable stabilization strategy plays a key role for the wide operation of these highly dynamic systems in the industrial sector. This study addresses a method in which mobile manipulators independently detect instabilities and trigger countermeasures to prevent them from tilting. A tip-over detection algorithm was implemented based on the Moment Height Stability method, whose main advantage is the examination of all dynamical influences affecting the mobile manipulator to indicate how stable/unstable the system is during its operation. Consequently, the theoretical workspace of the robot manipulator was redefined using a certain critical boundary surface subjected to the analysis of the stability value. This workspace optimization was complemented with an additional algorithm in which the robot manipulator joints adopt an appropriate configuration to compensate in real-time detected instabilities. During repositioning of the robot manipulator's arm, the initial orientation of its tool center point is maintained to avoid a workpiece mishandle. The simulation and experimental results suggest that the system is stable and safe to operate in changing environments, thus providing a universal approach to avoid the inherent stability problems of agile and lightweight mobile manipulators.

**Keywords:** mobile manipulators; Moment Height Stability method; repositioning the robot manipulator's arm; stabilization strategy; workspace optimization



**Citation:** Toledo Fuentes, A.; Kempf, F.; Kipfmüller, M.; Bergmann, T.; J. Prieto, M. Tip-Over Detection and Avoidance Algorithms as Stabilization Strategy for Small-Footprint and Lightweight Mobile Manipulators. *Machines* **2023**, *11*, 44. <https://doi.org/10.3390/machines11010044>

Academic Editor: Xinjun Liu

Received: 1 December 2022

Revised: 27 December 2022

Accepted: 28 December 2022

Published: 30 December 2022



**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Compact and lightweight mobile manipulators tend to tip over when the robot manipulator works under heavy loads, when its position comprises specific critical arm configurations, or when fast or abrupt movements are induced by the mobile platform [1–3]. Some severe implications for a mobile manipulator in terms of tip-over during normal operation include damage or loss of workpieces, damage to the robotic system itself and the surroundings, or even personal injury. Another important aftereffect of a tip-over, especially for mobile manipulators that are put in motion by differential drives, is the noncompliance between the tool center point (TCP) calculated by the robot and the real current value that has been altered by the loss of contact between the drive wheels and ground during the tip-over. Even for mobile manipulators equipped with a hydraulically sprung undercarriage, a dramatic compression/expansion of its suspension during an abrupt braking process can be counterproductive. Both circumstances cause inaccuracy in handling or in avoiding collisions with other objects that, according to the trajectory computations, should be clear of the path.

A system starts to tip over when applied forces generate a torque large enough to initiate rotational motion around the tilting edge. In this case, the stability moment is smaller than the tilting moment [4]. Therefore, to improve stability and prevent tip-over, the stability moment must be increased. Two different concepts have been used to produce a higher-stability moment for mobile systems:

1. The center of gravity (COG) of the system is shifted by attaching additional masses as counterweights.
2. The tip-over edge is shifted to increase the corresponding lever arm.

However, these two approaches have failed to address the loss of mobility caused by an increase in weight and dimensions. In the specific case of mobile manipulators, these approaches are unsuitable since the mobile platform has a restricted payload, and the heavier the payload, the less agile the system is (important for the estimation of cycle time) and the shorter the battery supply lasts. Therefore, it is evident that mobile manipulators require a special solution for their instability problem that does not restrict mobility and dexterity.

This study had two key objectives. Firstly, it deals with the recognition of instabilities using a tip-over detection algorithm that provides information about the stability of the system in relation to all possible tip-over edges of the mobile manipulator. Secondly, a tip-over avoidance algorithm that prevents the mobile manipulator from tip-over by repositioning the robot arm joints. A prerequisite for the tip-over avoidance is that the initial orientation of the manipulator TCP must be maintained throughout the repositioning process.

## 2. Literature Review

To date, several techniques have been developed to analyze the dynamic tilting stability of mobile systems. The study proposed by Ghasempoor and Sepehri [5] calculated the system energy level with respect to the tilt axis. On the other hand, Li [6] suggested the evaluation of the support forces between the wheels and the ground. A significant analysis and discussion of the subject was presented by Papadopoulos and Rey [1,7,8], the so-called Force Angle stability measurement (FA), which inspects the vector between the system resulting total force (measured by an inertial measurement unit) and the normal vector of each tilting axis. The result of FA is a binary value that distinguishes between stable and unstable, but lacks a statement about the degree of risk. Another relevant approach is the estimation of a stability value by the so-called Zero Moment Point (ZMP), which was first established by Vukobratovic et al. [9] and implemented by Sugano, Huang, and Kato [10,11]. The ZMP assumes that a system is stable if the sum of all moments acting on the system is equal to zero. Internal forces generated by the joints and their movements as well as external forces produced by the acceleration of the mobile platform [12] are also considered. The ZMP evaluates a system as stable if the cut-off of the resulting moment vector lies inside a tip-over polygon. Another well-known approach is the Moment Height Stability measure (MHS) introduced by Moosavian and Alipour [12–14] for computing the moments acting on each tilting axis of the system. In addition to the internal and external forces (as in ZMP), moments of inertia and their effects provide a stability value for each tip-over axis, indicating the critical tilt edge.

The effectiveness of the last three approaches has been evaluated and compared by other researchers. The authors in [14] confirmed that the ZMP method differs substantially from the other two methods because of the fact that ZMP does not directly consider the inertial moment of the bodies and the height of the COG. Furthermore, the FA method requires significantly more computing power than the ZMP and MHS method. In turn, the MHS demands much lower computing power and considers the system dynamics. Roan et al. [15] carried out comparisons using a real mobile robot system whose COG location was fixed during the entire set, obtaining similar results to those in [14].

Besides the tip-over detection, the supplementary aspect to solve the instability problem of mobile manipulators is the tip-over avoidance. Generally, two basic approaches are currently being adopted in researches for the tip-over avoidance in mobile manipulators:

- either the traveling speed of the mobile platform is reduced/suited, or
- the manipulator takes another position/orientation.

A combination of both approaches is also possible [1,3,16].

Particularly for the problem of estimating the new positions the manipulator should adopt to achieve stable operating states, existing countermeasures fall into the following categories: moving the robot arm to a predefined safe position, compensation of movement (action—reaction) or repositioning of robot arm based on control schemes such as fuzzy logic.

Perhaps the best-known studies include those carried out by Rey et al. [1], who proposed a strategy based on FA for tilting detection in such a way that, if an instability is detected, the manipulator should assume a predefined safe position [1,17]. However, the major disadvantage of this method is that the executed motion-planning task must be completely aborted for repositioning, leading to large delays during the operation time. In [18], the same authors detect instabilities by means of the so-called contact force method, using the same principle as in previous works (if the mobile manipulator is in a noncritical state, the robot arm is moved) but including the fact that the new joint arrangement of the manipulator depends on the system COG and its speed, and adopting the position in which the contact force is the same at all contact points (wheels). The limitation of this approach lies in the need to measure the contact forces between the wheels and the ground, which requires the use of additional sensors and their integration on the wheels. Hatano and Obara [19] employed a highly simplified mobile manipulator consisting of a single link with mass; as soon as an instability was detected by means of ZMP, the manipulator was moved to a predefined position, thus generating a force acting against the tilting moment. Ding et al.'s [3] algorithm detected the risk of tip-over using the Improved Tip-Over Moment Stability Criterion (ITOMSC). The position and orientation of the manipulator's joints should be adapted or the speed of the mobile platform should be assumed to accomplish the calculated optimum tilting moment for the current state of the mobile manipulator. Once again, this procedure forces the mobile manipulator to abort its original motion-planning task, standing still during stabilization, and resuming it after the system is considered stable. The new position/orientation of the joint of the manipulator does not have to be part of the original trajectory path. The algorithm presented by Huang and Sugano [16] calculates an unadjusted optimal trajectory, then inspects which points of the entire trajectory do not fulfill the ZMP criteria and recalculates these critical points as often as necessary until the criterion is fulfilled for each point. Only then is the mobile manipulator ready to operate. According to the authors in [16], a tilting risk by means of ZMP will theoretically never be detected during task execution if the manipulator follows the optimized pre-calculated path. This approach manipulates the motion and path planning of the robot arm enormously and, perhaps, the most serious disadvantage of this method is that it cannot react to unexpected behaviors (e.g., abrupt braking maneuvers during traveling). Furuno, Yamamoto, and Mohri [20] implemented a similar method based on the ZMP for tilting detection and a predefined stable trajectory path, which adjusts beforehand the position of the manipulator and movement of the mobile platform; the same limitations indicated above apply to this study. Kim et al. [21] suggested a real-time null-space motion approach that is applicable only to robots with kinematic redundancy. A different real-time method was proposed by Li and Liu [22,23] as well as by Alipour et al. [17], who implemented fuzzy logic to obtain nonnumeric statements that depend on the joint positions, velocities, and accelerations. Alipour et al. [17] triggered a fuzzy logic algorithm according to the information delivered by the MHS for tip-over detection, whereas Li and Liu [22,23] relied on the contact forces between the wheels and the ground.

All these approaches ignore either the real-time aspect, in order to react against unexpected abrupt braking maneuvers, the numerical statements (and not just Boolean) in order to detect the degree of tip-over risk, or the simplicity of its measurement methods.

Therefore, the central thesis of this work was to develop an active stabilization strategy that addresses the following features:

- The mobile manipulator should detect any degree of instability even under dynamic conditions.

- Appropriate and simplified measuring instruments shall be used.
- The mobile manipulator should not abort the tasks currently being executed, thus avoiding wasting time during its operation.
- The motion control should not be manipulated for these purposes; it should only be used to provide target positions and orientation values.
- The motion path should not be predefined, but updated with the corrected repositioning path in real time depending on the stability values calculated by the MHS. The repositioning path must remain close to the original path and the original orientation of the TCP should be maintained.

A simple but effective method defined by a gradient-based motion planning subject to a distance minima has been integrated in the teleoperation algorithm for the Robot-Assisted Minimal Invasive Surgery [24]. Gradient-based planning approaches have also been implemented for the calculation of collision-free trajectories for robot arms and quadruped robots [25,26], with the collision positions being the constraints for the gradient functions. Another similar, well-known method to address the real time collision avoidance problem for manipulators and mobile robots is the Artificial Potential Field (APF), established by Khatib [27]. APF functions are implemented to design nonlinear control laws based on energy gradient, e.g., using Lyapunov-based control scheme (LbCS).

Gradient-based approaches resemble the APF method, as both attempt to obtain local optimum for real-time low level control (execution of specified simple path operations). Then, employing this local optimum and considering the operational space of the manipulator, high level planning algorithms calculate optimal trajectories to a designated target (global optimum). The potential field of the APF is formulated with functions that describe the energy of the system. It defines the target position for the end-effector as attractive pole, and obstacles as repulsive fields. Following this analogy, gradient-based functions could help to estimate a repositioning path for the manipulator addressed in this work if, instead of considering obstacles, the cost of trajectory is defined based on stability values of the mobile manipulator. Then, the computed repositioning points (local minima) could serve as input for the existing high level robot motion control to execute a repositioning path. Similar to the potential field to determine a collision-free path, to achieve stable position of the mobile manipulator the repositioning path can be attained by associating all consecutive repositioning spots (absolute minima) of the gradient-based function. Researchers have demonstrated the effectiveness of the APF method for mobile robots, such as quadrotors [28,29], ensuring their self-stabilization (no tilting, i.e., maintaining its orientation while in motion) during aerial maneuvers.

The proposed stabilization strategy was based on the Robot Operating System (ROS) framework and its effectiveness was empirically demonstrated on a real autonomous mobile manipulator.

### 3. Materials and Methods

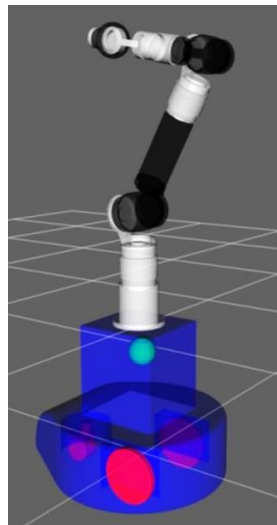
The data required for the development of the stabilization strategy were collected using the compact nonholonomic autonomous mobile platform Scitos G5 by MetraLabs and, as a handling tool, the 7-degrees of freedom (DOF) lightweight articulated robot arm LWA 4D by Schunk. The stability problem addressed in this study was observed in the proposed testing system during some experimental attempts: large forces and moments generated by certain configurations and/or motions of the robotic arm tended to tip the whole robotic system over.

Scitos G5 uses a differential drive to move. As already mentioned, direct contact between the ground and its two drive wheels is essential; otherwise, if the drive wheels raise off the ground, the theoretical current location of the TCP in the real-world space will be incorrect because it will be calculated using the wrong parameters for its position and orientation.

Since the mobile platform Scitos G5 is controlled by the Middleware for Robotic Applications (MIRA) [30] and the 7-DOF articulated robot LWA 4D is controlled by the Mid-

deaware ROS, a software/firmware integration was needed in addition to the mechanical and electrical integration. Because of the easiness ROS involves, an interface between MIRA and ROS should enable them to collaborate in ROS environments as follows: the mobile platform motion planning is calculated by MIRA, but its execution should be performed by ROS, acting as a single system with the articulated robotic arm. For the integration, the Spatio-Temporal Representation and Activities for Cognitive Control in Long-Term Scenarios (STRANDS [31]) was implemented in Scitos G5 as interface between MIRA and ROS. Hence, the integrated mobile manipulator can be controlled entirely using ROS.

The control system of the LWA 4D was prepared according to the already available software package developed by the Fraunhofer Institute for Manufacturing Engineering and Automation (IPA) as part of various cooperative research projects with the robot manufacturer Schunk [32–34]. Furthermore, the CAD model of the Scitos-Schunk-Mobile manipulator was adapted and validated ([35]) in order to obtain realistic kinematic and dynamic behavior of the system for simulation purposes. ROS visualization (RVIZ), for the 3D representation of the robotic system, the Whole Body Operational Space (WOBSC) [36] in MoveIt, as a control tool, and Gazebo, for the virtual world, assist the development of the algorithms in a close-to-reality virtual environment. Together, they provided the same functionalities as the real mobile manipulator, including the data transfer for the differential drives (servo motors) and the laser scanner detector (standard laser scanner [37]). Additionally, in order to detect tilting instabilities using the algorithm presented in the following sections, an inertial measurement unit (IMU [38]) was attached to the mobile platform. The integration of the main subsystems, robot manipulator and mobile platform, in the ROS environment is displayed in Figure 1.



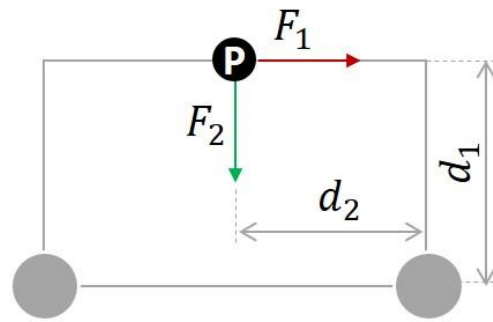
**Figure 1.** The virtual model of the coupled robotic systems.

### 3.1. Tip-Over Detection Algorithm

#### MHS Algorithm Adaption and Implementation

The tip-over detection algorithm should consider all forces and torques generated by the accelerations and inertia forces of the system itself acting on a tip-over edge. Since the FA method demands more computing power than the MHS method and, in addition, MHS considers dynamical influences in the calculations, MHS is employed in this study for further tip-over detection.

The MHS computes the equidistant and orthogonal forces ( $F_1$  and  $F_2$  in Figure 2) that affect the system stability and their vertical and horizontal distances to the tip-over axis ( $d_1$  and  $d_2$  in Figure 2).



**Figure 2.** Forces and distances employed in MHS calculations (based on [12]).

In the example in Figure 2, considering the right wheel as tilting point, the greater the ratio  $F_2 \cdot d_2$  to  $F_1 \cdot d_1$ , the higher the stability.

System stability disturbances appear when the sum of the tilting moment around one or more of its tilting edges,  $M_{\text{tilting}}$ , is greater than the sum of its stability moment around the same edge(s),  $M_{\text{stability}}$ , as described in (1).

$$S = \sum M_{\text{stability}} / \sum M_{\text{tilting}} \quad (1)$$

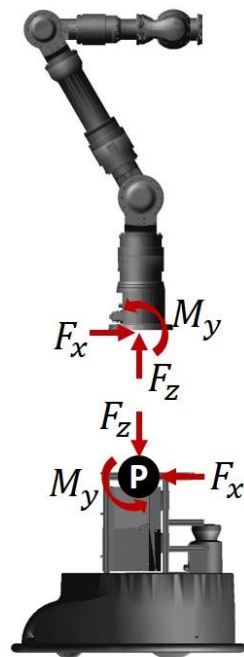
For,

$S \geq 1$ , the system is stable and balanced,

$S = 1$ , limit value to define a system as stable or not (the body is not yet tilted),

$S < 1$ , the body is unstable and tends to tip over.

For the estimation of the tilting moment, all joint positions, joint angular velocities, and joint angular accelerations of the manipulator over time need to be determined. The inverse dynamics of the mobile manipulator provided the basis for calculating the MHS. It includes all 6-DOF force and torque components acting on P (see Figure 3), where the manipulator and mobile platform are physically connected. The components of  $\vec{F}$  and  $\vec{M}$  represent how the manipulator affects the mobile platform in terms of the forces (including inertial) and torques generated only by the movement of the manipulator itself.



**Figure 3.** Connection point P of the mobile platform with the manipulator.

The development was performed in ROS simulation environments. For this reason, the simulation model shown in Figure 3 was incorporated with an equivalent 6-DOF massless fix joint at point P to emulate the physical joint; otherwise, the forces and moments acting on the mobile platform across connection point P could not be estimated.

One of the best-known methods for determining the inverse dynamics of a robot considered as a rigid body is the Recursive formula of the Newton-Euler Algorithm (RNEA) implemented by Featherstone [39]. The RNEA requires all forces and torques acting on each joint, the system geometrical characteristics, and the parameters regarding both the kinematics, such as joint position ( $S_i$ ), joint velocity ( $v_i$ ), and acceleration ( $a_i$ ), as well as the dynamics, such as inertia ( $I_i$ ), mass ( $f$ ), and center of mass ( $f_i^B$ ). All 6D spatial vectors employed in the RNEA ( $v_i$ ,  $a_i$ ,  $S_i$ ,  $I_i$ ,  $f$ ,  $f_i^B$ ) are described for the coordinate systems of each element of the kinematic chain of the robot manipulator. In ROS, via the topic "joint\_states", the joint positions and joint velocities of the robot manipulator were read out from its actuators, but not its joint accelerations. Therefore, the algorithm in ROS was complemented with a class named "acceleration\_publisher", which calculates the derivative of the joint velocity with respect to time. The rest of the parameters (inertia, mass and center of mass) were determined using the CAD volume model of the mobile manipulator.

The RNEA algorithm developed to solve the inverse dynamics based on Featherstone's statements has already been implemented by Smits [40,41] in ROS. Therefore, it had just to be adjusted for the employed testing system. The algorithm first determines the velocity and acceleration of all links, beginning from the basis (0) to the end effector (TPC). Then, the forces and torques acting on each link  $i$ , as a result of the accelerations, are calculated by combining equations for the linear and rotational motions of rigid bodies given by Newton and Euler.

The resulting moments acting around the arm joints obtained from the topic "rne\_return" were validated for plausibility on the real mobile manipulator using the following two scenarios:

1. The COG of the 5th, 6th, and 7th links of the manipulator were vertically aligned close to the rotary axis of its 4th joint so that no extra moments were produced around the 4th joint axis. While the 4th joint was accelerated from this vertical position to the sides, the moment acting on the 4th joint ( $M_4$ ) was continuously calculated using the inertia,  $I$ , and the angular velocity of the 4th joint axis,  $\omega_4$ , as in (2).

$$M_4 = I \cdot \omega_4 \quad (2)$$

Additionally, the COG of the 4th link is shifted during the movement, and thus, an extra moment is generated from the gravity vector and the added lever arm. Table 1 lists the results obtained from analytical calculations and from the output of the RNEA algorithm.

2. The manipulator's arm is set vertically. Now, instead of accelerating its 4th joint (as in scenario 1), the mobile platform is linearly accelerated while the manipulator remains stationary with

$$\vec{a} = \begin{pmatrix} 0.37 \\ -0.03 \\ -0.13 \end{pmatrix} \text{m/s}^2, \quad \vec{S} = \begin{pmatrix} -0.0021 \\ -0.0017 \\ 0.6307 \end{pmatrix} \text{m}$$

where  $\vec{a}$  describes the linear acceleration of the mobile manipulator and  $\vec{S}$  defines the position of its COG related to the point P, illustrated in Figure 3.

**Table 1.** Torque generated at 4th joint.

Analytical	RNEA Using Real Robot
0.49 N·m	0.49 N·m

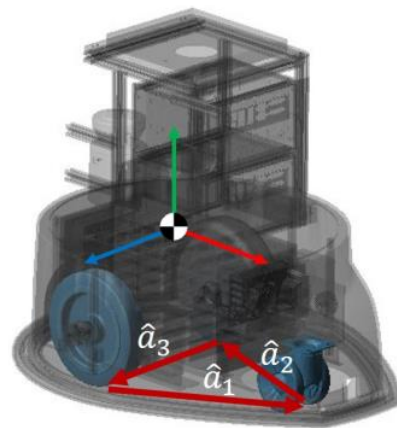
The inertial forces generated by the acceleration were calculated using D'Alembert's principle. Table 2 lists the forces and torques at the connection point P (see Figure 3), thus validating the RNEA algorithm.

**Table 2.** Forces and torques generated at connection point P.

Force/Torque Components	Analytical	RNEA Using Real Robot
$F_x$	8.5 N	8.5 N
$F_y$	-3.2 N	-3.2 N
$F_z$	196.0 N	196.0 N
$M_x$	1.7 N·m	1.7 N·m
$M_y$	5.8 N·m	5.8 N·m
$M_z$	0.0 N·m	0.0 N·m

Based on the outcomes comprising the inverse dynamics as well as on the measurements from the IMU and the location of the COG related to the mobile coordinate system, the dynamic stability could be estimated using the MHS.

For the testing system presented in this work, the three tip-over tilting axes were defined by the three unit vectors added to close the shape built by the wheels of the mobile platform, as shown in Figure 4.



**Figure 4.** Tilting shape and platform COG.

As shown in (3), the resulting forces  $\vec{F}_r$  and moments  $\vec{M}_r$  on the massless connection point P (Figure 3) represent their components acting on the position vector  $\vec{p}_i$  derived from both subsystems, the mobile platform and the manipulator.

$$\vec{M}_{vi} = -\vec{p}_i \times \vec{F}_r + \vec{M}_r \quad (3)$$

These moments  $\vec{M}_{vi}$  acting on the vertices can be projected onto the edges of the tilting shape by means of a scalar product between them and the tilting edge  $\hat{a}_i$ , as in (4). The result is the unit vector  $M_{ei}$  that represents the projection of the tilting moment on the edge  $i$ .

$$\hat{M}_{ei} = \vec{M}_{vi} \cdot \hat{a}_i \quad (4)$$

The dynamic stability value  $\alpha_{ei}$  related to edge  $i$  can be estimated by considering the moment of inertia of the mobile platform,  $I_{ei}$ , with respect to the corresponding edge  $i$

$$\alpha_{ei} = (I_{ei})^{\sigma_i} \cdot \hat{M}_{ei} \quad (5)$$



where

$$\sigma_i = \begin{cases} +1 & \text{for } M_i > 0 \\ -1 & \text{for any other case} \end{cases}$$

Additionally,  $\alpha_{ei}$  should be very sensitive to the height of the COG system related to the ground,  $h_{COG}$ . Thus, the equation can be complemented as follows:

$$\alpha_{cm-i} = (h_{COG})^\lambda \cdot \min(\alpha_{ei}) \quad (6)$$

with

$$\lambda = \begin{cases} -1 & \text{for } \min(\alpha_{ei}) > 0 \\ +1 & \text{for any other case} \end{cases}$$

The MHS defines the critical value as the smallest dynamic stability value  $\alpha$  regarding the edge of polygon  $i$  :

$$\alpha = \min(\alpha_{cm-i}) \quad (7)$$

and can be interpreted as

$\alpha > 0$  system is stable,

$\alpha = 0$  system is critically stable,

$\alpha < 0$  system tends to tip over edge  $i$  from  $\min(\alpha_{cm-i})$ .

### 3.2. Tip-Over Avoidance Algorithm

Once the “tilt\_detection” ROS node is able to identify a possible overturn of the robotic system, the next step is to react actively against these instabilities by means of a tip-over avoidance algorithm.

As mentioned in the literature review, there are two fundamental procedures for solving the problem of tip-over avoidance for mobile manipulators: the traveling speed of the mobile platform is reduced or the manipulator takes another configuration in space. Considering that a reduction in the traveling speed of the mobile platform involves cycle time loss, this study set out to implement the repositioning of the manipulator joints by retaining the TCP orientation as probably the best and most efficient method to avoid a tip-over of the mobile manipulator.

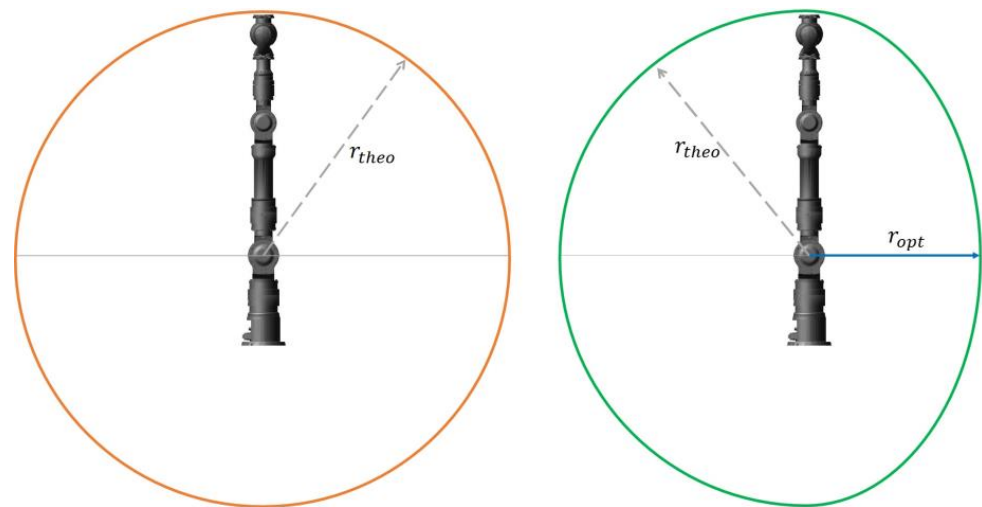
Therefore, in order to properly shift the COG of the manipulator, the joints of the manipulator should be repositioned in relation to the tilt stability value around the tilting edges calculated by the MHS. This increases the distance between the tilting edge and the system COG so that the value from the MHS indicates a more stable state. If the stability value resulting from the tip-over detection algorithm lies below a predefined stability threshold, a repositioning of the robot arm should be triggered. The strategy to avoid tip-over must be designed such that the robot takes the shortest possible time for the overall task.

The mobile manipulator becomes unstable and starts to tip over under the following scenarios:

- When one of the subsystems, the base or robotic arm, is stationary, while the other subsystem is moving.
- When the mobile platform and the robot arm move simultaneously.

Therefore, the algorithm has to consider both a predictive (repositioning before the mobile platform moves) and a real-time (repositioning during travel) countermeasure.

The theoretical local workspace, which was originally described as a sphere with its center at the second joint of the robot arm (see Figure 5a), cannot be fully deployed owing to the tip-over problem: The farther the robot arm TCP is moved from its center axis within the theoretical local workspace, the more unstable the system is with respect to the two front wheels.



**Figure 5.** Theoretical (a) and noncritical workspace optimization (b).

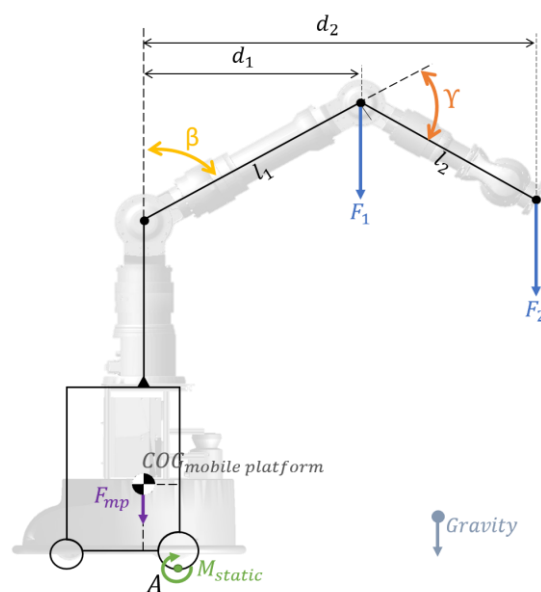
According to this fact, a re-definition of the theoretical local workspace can be inferred by finding an as-large-as-possible optimum volume so that the robot TCP has maximum room to move freely and safely without the risk of tip-over. In addition, the configuration space of the robot must be adapted so that the robot arm does not move beyond the limits of the optimized workspace, leading to a tip-over too.

For this purpose, the entire theoretical workspace was divided into two parts, as illustrated by the green area in Figure 5b:

1. A noncritical area described by a hemisphere with the maximum radius of the original theoretical workspace, and
2. A critical workspace designed as an ellipsoid, since its volume builds the sphere very closely.

The appropriate dimensions of the ellipsoid are defined by considering the tilting moment  $M_{static}$  with respect to point A (see Figure 6), as in (8).

$$M_{static} = \sum_{i=0}^n F_i \cdot d_i \quad (8)$$



**Figure 6.** Free body diagram of mobile manipulator.

As shown in Figure 6, since the lever arm of forces  $F_1$  and  $F_2$  to the center of rotation A changes, the moment generated by force  $F_1$  depends on the angle  $\beta$  of the second joint of the robot arm. Likewise, the lever arm for force  $F_2$  is determined by  $\beta$  and  $\gamma$ , as follows:

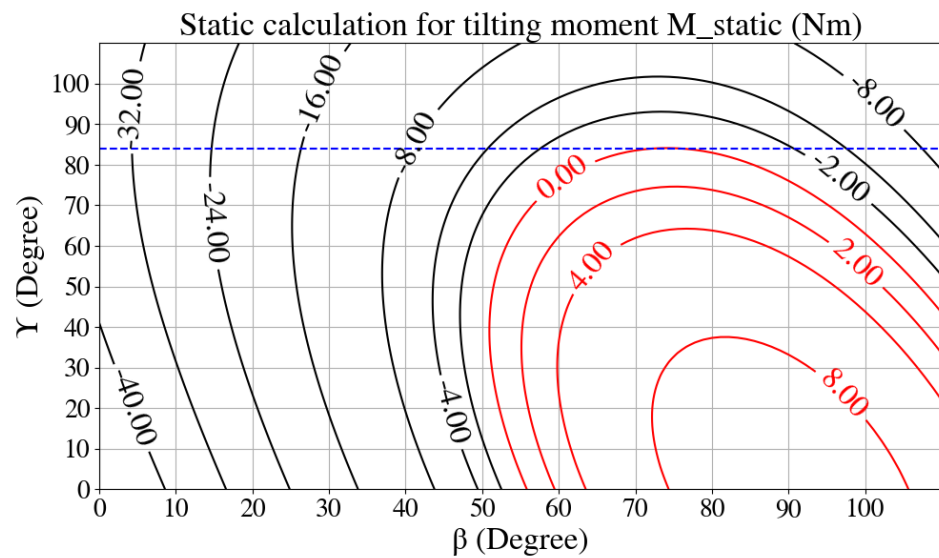
For  $F_1$ ,

$$d_1 = l_1 \sin(\beta) \tag{9}$$

For  $F_2$ ,

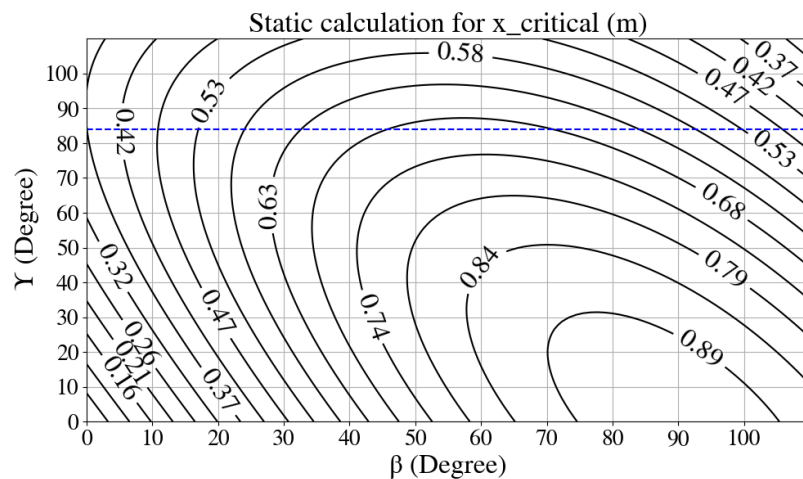
$$d_2 = l_1 \sin(\beta) + l_2 \sin(\beta + \gamma) \tag{10}$$

The diagram in Figure 7 shows the static tilting moment,  $M_{static}$ , as a function of a wide range of values for angles  $\beta$  and  $\gamma$ . The red lines indicate the critical range at which the system was unstable.



**Figure 7.** Estimation of tilting moment under static conditions for different angle configurations  $\beta$  and  $\gamma$ , based on the parameters obtained from CAD volume model of the mobile manipulator.

The minimum value of  $\gamma$  for which the calculated tilting moment does not exceed 0 N·m over the entire interval is indicated by the blue auxiliary line at  $\gamma = 84^\circ$  in Figure 7. Based on this value, the maximum semi-minor axis for the semi-ellipsoid can be deduced in Figure 8 to be approximately 0.7 m, which is the greatest possible arc that guarantees stability.



**Figure 8.** Estimation of the ellipsoid critical radius under static conditions for different angle configurations  $\beta$  and  $\gamma$ , based on the parameters obtained from CAD volume model of the mobile manipulator.

The implementation of this semi-ellipsoid volume is illustrated in Figure 9. If the mobile platform is not moving, only within this optimized workspace can the manipulator extend its links as far as possible without causing instability. All joints and links of the robot arm maintain this optimized workspace when moving to the target coordinates.

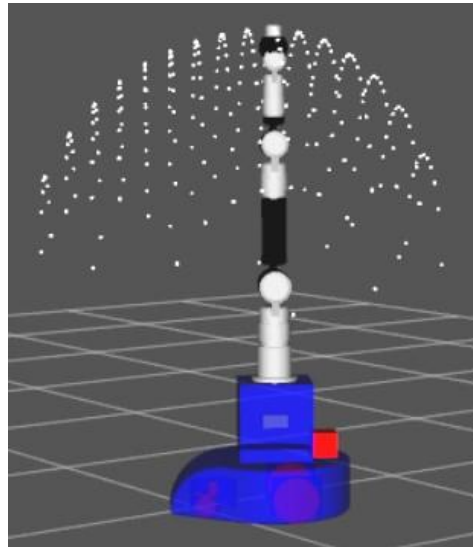


Figure 9. Optimized workspace for robot arm in RVIZ.

Once the optimized workspace was defined, it was also necessary to determine whether the robot TCP was located within the predefined critical area (in front of the robot) or within the non-critical area (behind the robot). For this purpose, the vector between the mobile platform and the robot TCP was calculated and transformed into the coordinate system of the mobile platform.

As shown in Figure 10, the scalar product between the direction vector  $\vec{x}_{mp}$  and vector  $\vec{TCP}_{mp}$  can be used to infer the value of angle  $\Gamma$  as shown in (11).

$$\Gamma = \cos^{-1} \frac{\vec{TCP}_{mp} \cdot \vec{x}_{mp}}{\|\vec{TCP}_{mp}\|} \quad (11)$$

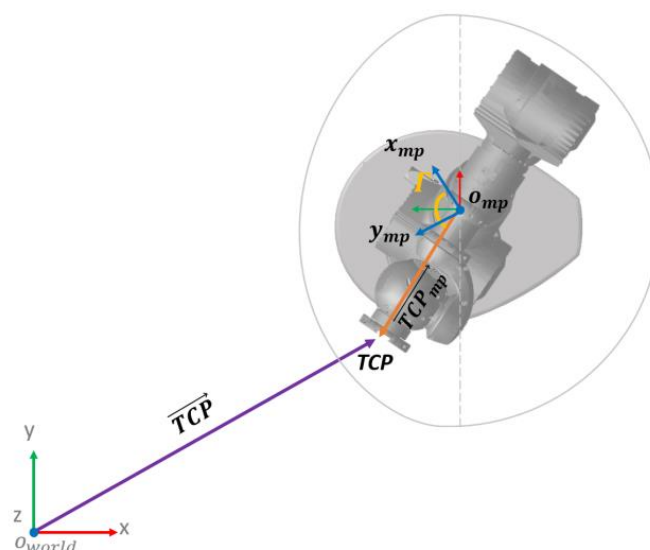


Figure 10. Angle  $\Gamma$  defines if the TPC is located within the critical or noncritical volume.

The angle  $\Gamma$  provides information about the position of the TCP relative to the mobile platform. In this respect, it can be determined whether the equation for the hemisphere or ellipsoid should be used for the operation. In general,

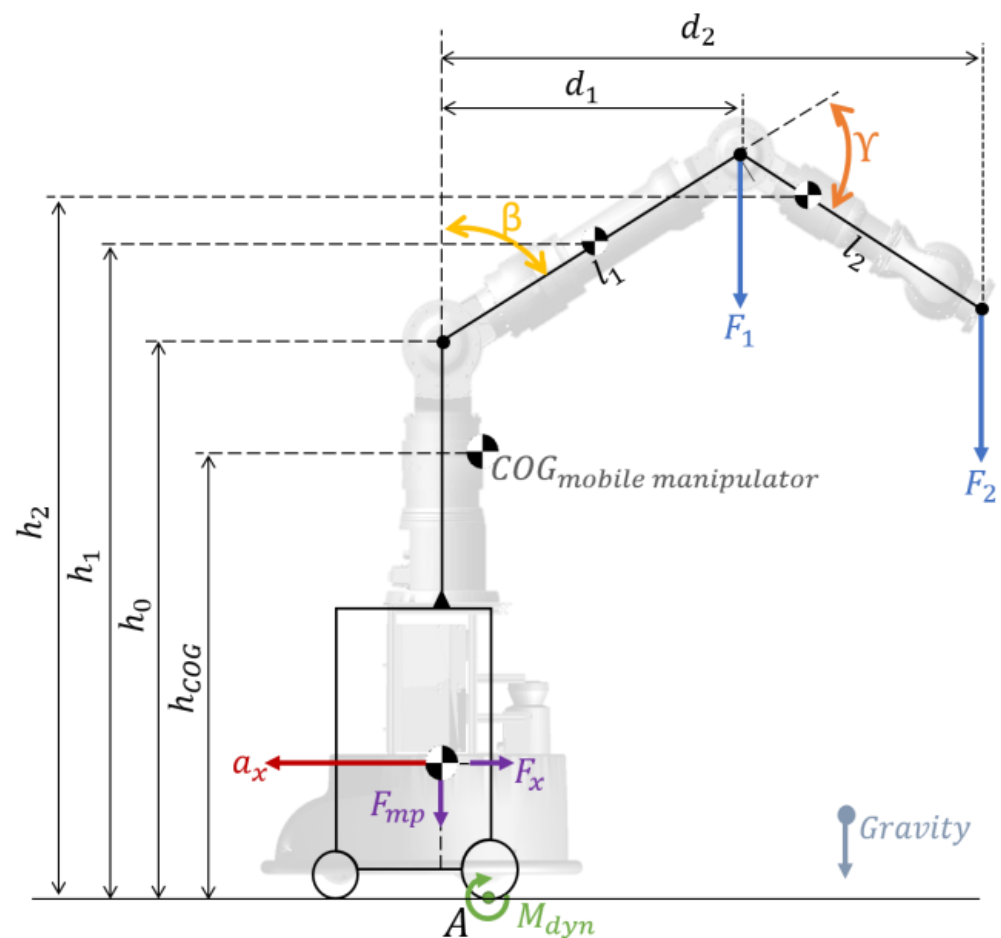
- When  $\Gamma < 90^\circ$ , the TPC is located within the critical area, and thus, the ellipsoid should be employed as permitted workspace for the motion planning.
- When  $\Gamma > 90^\circ$ , the TPC is located within the noncritical area; thus, the hemisphere defined by the theoretical local workspace should be employed for the motion planning.

An additional relevant circumstance to be examined is when both the manipulator and the platform are moving simultaneously. In this case, dynamic forces and moments are generated by the robot arm joints or the platform movement itself. Consequently, the development of a real-time active stabilization strategy must be considered: as soon as an instability is detected by the tip-over detection algorithm, the robot arm should be retracted into a stable position, while the orientation of its TCP should be maintained. A prerequisite is that the robot arm should move as little as possible during repositioning to avoid unnecessary deviations from the original path.

A similar analysis can be performed to determine the dynamic tilting moment resulting in

$$M_{\text{dyn}} = \sum_{i=1}^n F_i \cdot d_i - \|\vec{a}\| \cdot m_{\text{total}} \cdot h_{\text{COG}} \quad (12)$$

The height of the COG of the individual components  $h_i$ , depends on the angles  $\beta$  and  $\gamma$ , as shown in Figure 11.



**Figure 11.** Free body diagram of mobile manipulator considering dynamic forces and moments generated by operation movements.

The diagrams in Figure 12 show the tilting movement generated on the mobile manipulator under different accelerations of the mobile platform: The black level lines display all states in which the stability moment was bigger than the tilting moment (stable states). The red lines indicate the area where the tilting moment had a higher value than the stability moment (unstable states).

The area that represents the stable states is reduced along with the increasing acceleration of the mobile platform, because the inertia force depends proportionally on this acceleration. However, from the graph below, in Figure 13, it can be observed that the stability value  $\alpha_{cm-critical}$  is more dependent on the acceleration of the mobile platform than on the joint angle  $\gamma$  of the manipulator.

Considering the maximal acceleration of the mobile platform as  $0.8 \text{ m/s}^2$ , the critical stability value  $\alpha_{cm-critical} = 250$  was chosen. Using this value as a threshold for tip-over detection under dynamic loads, it can be ensured that the mobile manipulator does not tip over at any time during traveling.

Changes in dynamic tilting stability value can be studied in relation to the robot arm joint variables,  $q_i$ . By applying the gradient method into the calculations, it is possible to determine the direction in which the vector attached to the TCP should be shifted to achieve higher tilting stability. In addition, the time lost due to repositioning could be kept as low as possible. For this gradient function, no extra real-time capability is required; therefore, the available hardware and software can be used.

Spatial discretization was carried out to simplify the approximation of a related gradient method as a function of the current tilting stability value. For the discretization, revolute joints  $q_1$  to  $q_4$  of the robot arm were used because they have the greatest influence on the motion of the entire system.

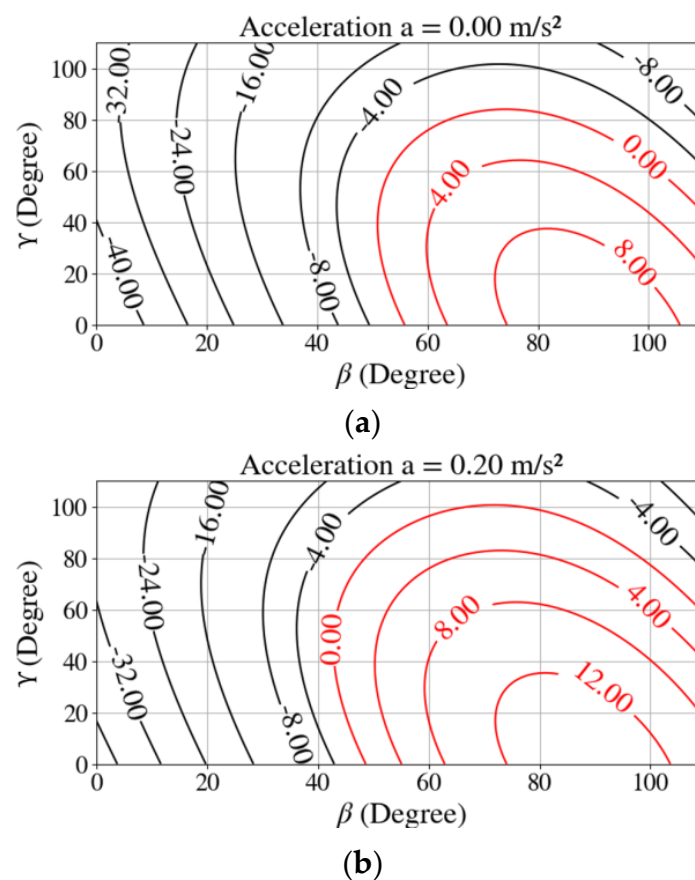
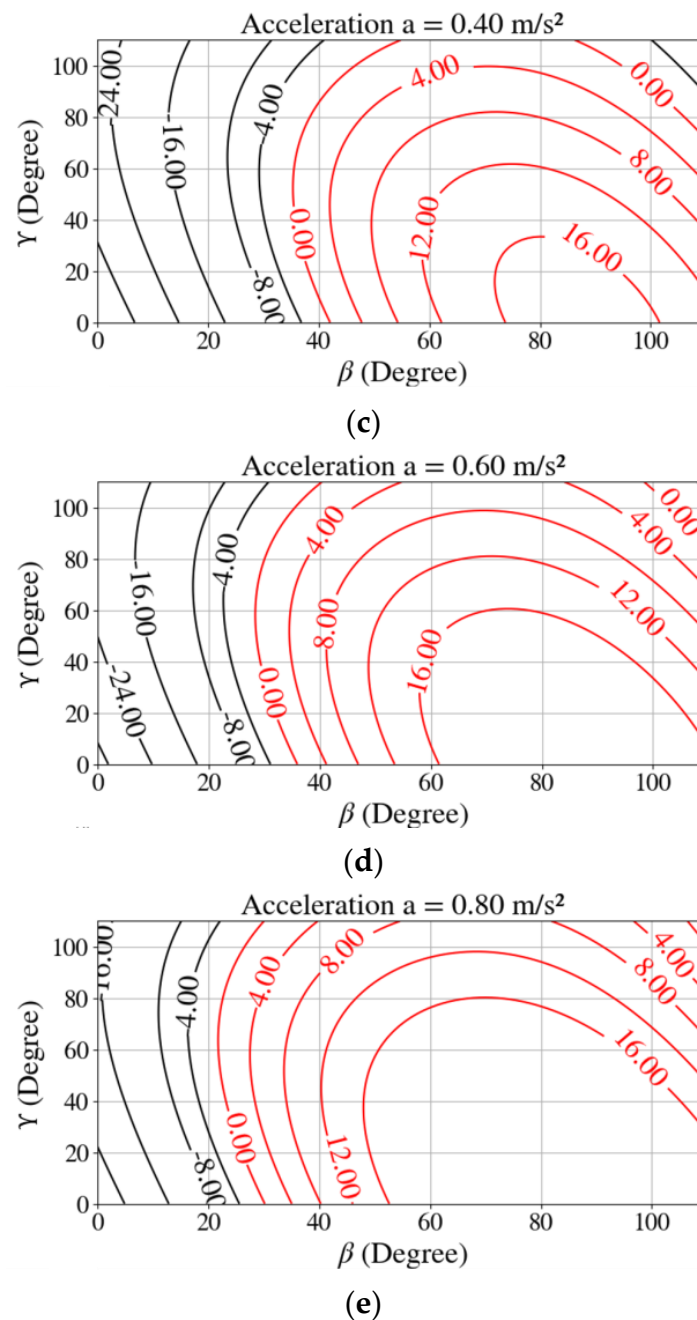


Figure 12. Cont.



**Figure 12.** Analysis of the stability moment as a function of the manipulator's joint positions and mobile platform acceleration, employing as linear acceleration: (a)  $0 \text{ m/s}^2$ , (b)  $0.2 \text{ m/s}^2$ , (c)  $0.4 \text{ m/s}^2$ , (d)  $0.6 \text{ m/s}^2$ , and (e)  $0.8 \text{ m/s}^2$ . The black lines represent stable states and the red lines represent unstable states of the mobile manipulator.

The red point in Figure 14 represents the current position of the TCP of the manipulator. Based on this configuration, the algorithm calculates the tilting stability value  $\alpha$  for the joint configuration that corresponds to each of the neighbor blue points. If the maximum value determined for the tilting stability,  $\alpha_{\Delta \pm q}$ , is larger than the threshold value,  $\alpha_{\text{cm-critical}}$ , then the new joint position is sent to the motion control and the manipulator adopts this position to improve its stability status. Otherwise, the algorithm iteratively calculates the value  $\alpha_{\Delta \pm q}$  until it fulfills the stability criterion. A graphical representation of this algorithm is shown in Figure 15.

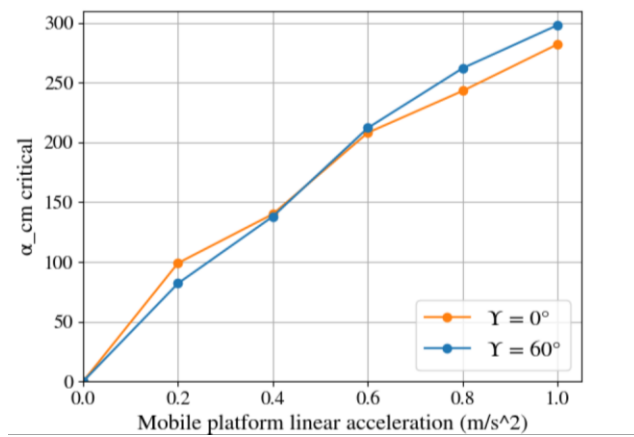


Figure 13. Critical tilting instability for different values of  $\gamma$  as a function of the mobile platform traveling acceleration.

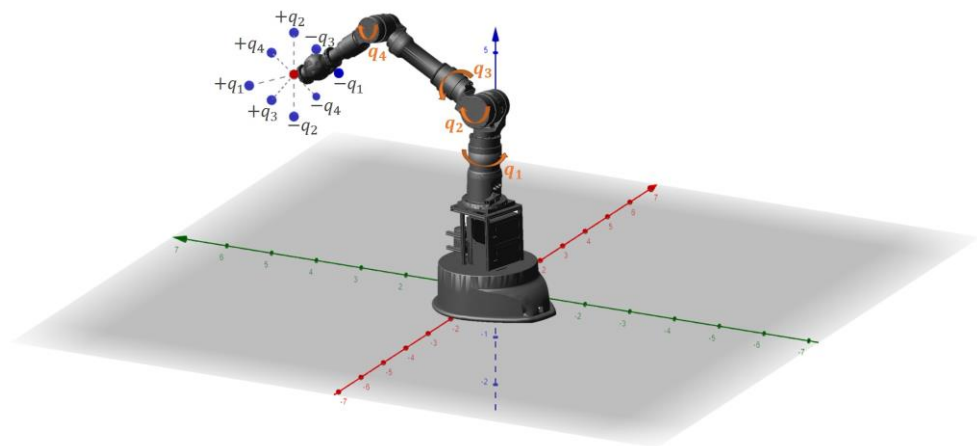


Figure 14. Discretization of workspace for the implementation of the gradient function.

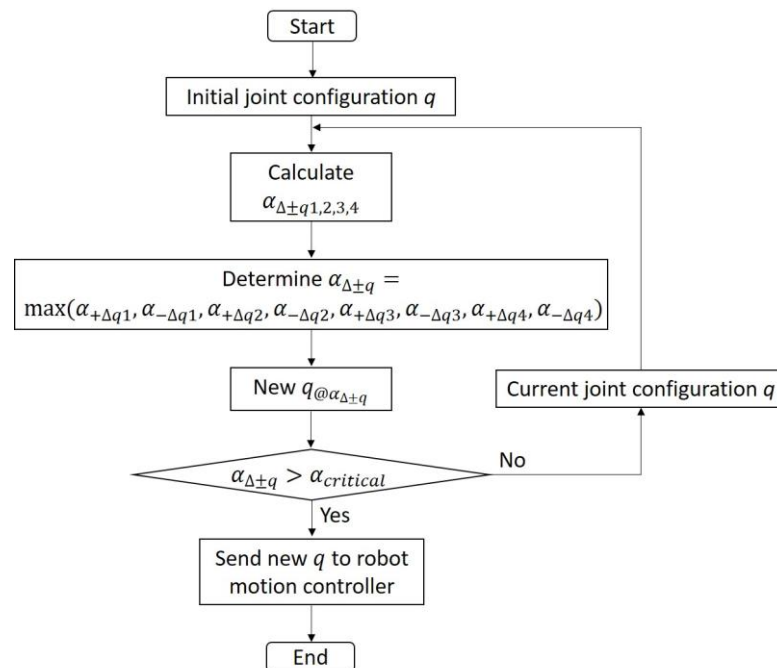
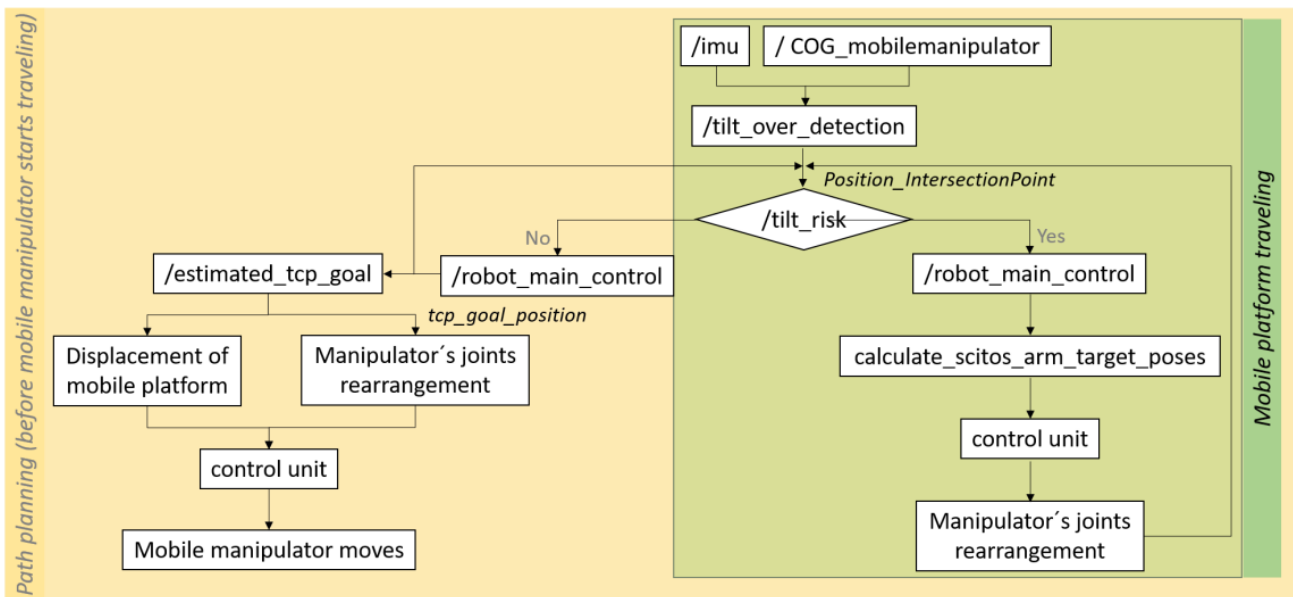


Figure 15. Algorithm for calculating the new joint position for the manipulator.



This gradient function was also implemented in ROS (see Supplementary Material): it determines the new TCP position for the robot arm with a higher tilting stability and returns it to the main program. Subsequently, the “SetPoseTarget” function relocates the corresponding joints within the collision-free optimized workspace, specifying the target orientation as the current orientation of the TCP. Owing to its high success rate and fast computation, the solver Rapidly exploring Random Tree (RRT) was implemented to search for a collision-free path within the configuration space [42,43]. Figure 16 illustrates the implemented control procedure.



**Figure 16.** Control procedure for stabilization strategy.

Before the mobile manipulator starts moving, the “/tilt\_over\_detection” node subscribes the topic of the IMU measurements, to get the data provided by the inertial sensors, and the topic COG, to get the position and orientation of the mobile manipulator’s COG. Once the required information is received, the same node “/tilt\_over\_detection” estimates the vector “Position\_IntersectionPoint” for all tilting edges and, consequently, publishes this information in the topic “/tilt\_risk”. If the Position Intersection Point delivers a positive value, the calculated force points outside the overturning shape and the mobile manipulator is prone to tip over. Following this, the node “/robot\_main\_control” subscribes the information contained in the topic “/tilt\_risk” and controls the robot manipulator to move to a safety position within the predefined optimized workspace, employing the gradient approach. After the mobile manipulator is considered as stable, the topic “/estimated\_tcp\_goal” sets and publishes how to reach the desired tcp goal position, either only performing a rearrangement of the robot manipulator joints or including the displacement of mobile platform to another point in the space. After the path planning is completed, the target position and orientation for each robotic subsystem are posted iteratively to the control unit and the mobile manipulator starts moving to the goal position. Meanwhile, the “/tilt\_over\_detection” checks the current stability value  $\alpha_{(cm-i)}$  continuously: as soon as  $\alpha_{(cm-i)}$  drops below the predefined critical value, the node “/robot\_main\_control” starts the repositioning, since the whole system is unstable. Once the system exhibits a stable state, the “/estimated\_tcp\_goal” temporarily adopts the new position (after repositioning) in the path planning as target position for the robot manipulator until the mobile platform reaches its target position, so as to not affect the stability state achieved.

## 4. Results

The presented stabilization strategy was verified, first, by means of simulations and, then, by employing the real robot under the following considerations:

- The configuration space was spatially discretized with  $q = 0.1$  rad for the manipulator joints  $q_1$  to  $q_4$  to simplify the implementation of the gradient approach.
- The robot workspace was set as defined in the previous section.

### 4.1. Evaluation of Workspace Optimization

When the mobile platform did not exhibit any accelerations (static tilting stability value) during the simulations, the semi-ellipsoid from the optimized workspace was set to  $r_{opt} \leq 0.8$  m. Above this value, the system started to tip over. The MHS exhibited a dynamic tilting stability of approximately  $\alpha_{critical} = 0$ . On the other hand, the semi-ellipsoid on the real robot could be increased up to  $r_{opt} \leq 0.9$  m. Around this value, the mobile manipulator presented evident strong vibrations, being an indication of a tip-over. The MHS exhibited dynamic tilting stability of approximately  $\alpha_{cm-critical} = -150$ .

Before the tip-over avoidance algorithm was considered, further empirical procedures were required to fine-tune the value obtained by the developed tip-over detection algorithm for the real robot. This requires the use of an offset that serves as a calibration parameter for the instability trigger point  $\alpha_{cm-critical}$ . The small deviation of the braking process in the simulation (<5%) causes the tilting stability in the real robot to be higher than that in the simulations. Another possible influence of this divergence in  $\alpha_{cm-critical}$  is the discrepancy between the real robot arm and the real mobile platform structure, and the robot description used for calculation (Unified Robot Description Format): Some connecting elements, cables, and brackets that are mounted on the real robot were not included in the robot description. With this calibration procedure, it can be guaranteed that the tip-over avoidance algorithm also exhibits comparable behavior using the real mobile manipulator.

### 4.2. Evaluation of Stabilization Strategy by Means of Simulation Sets

The entire stabilization strategy is considered as successfully validated if the following requirements are fulfilled:

- Optimized workspace: The mobile manipulator is capable of deciding if the mobile platform has to be moved to reach the target point without any risk of instability.
- In the case of a repositioning of the robot arm, the initial orientation of the TCP should be maintained.

To assess the stabilization strategy, the following simulation sets were performed in Gazebo first without the effect of the stabilization strategy and, then, repeated under the same conditions with the active stabilization strategy.

The robot description of the mobile manipulator, with their respective physical properties (such as geometry, center of gravity, inertia, etc.), was imported in Gazebo as URDF-File. The IMU and scanners of the mobile manipulator were also integrated in the simulations as ROS nodes. The mobile manipulator was placed in a previously prepared map (incl. environment such as obstacles) in known different positions and orientations before it starts to move towards the target point, located outside the optimized workspace. As with the real robot, the mobile robot in the simulations is controlled by the same motion planning algorithm [32–34,42,43] and finds its orientation by comparing the information from the scanner with the map. Information available from the actuators of the mobile manipulator (joint position and joint velocity) is extracted using ROS nodes [32–34]. The stability value is delivered by the MHS method, implemented in ROS in this work.

The performed sets and their corresponding outcomes are presented in Table 3.

### 4.3. Stabilization Strategy Using the Real Mobile Manipulator

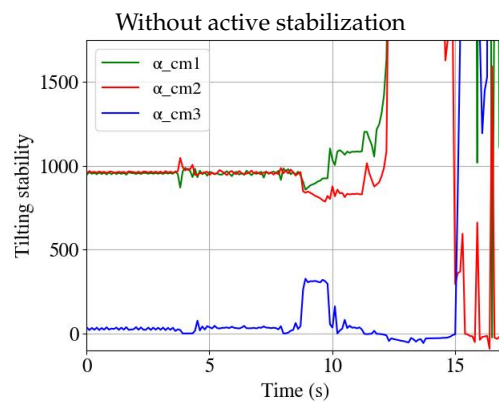
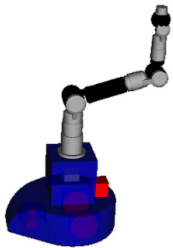
In the following scenarios, a tip-over of the real mobile manipulator was not induced to avoid damage to the equipment.

In the robotic lab, the mobile platform was assigned to an arbitrarily known origin coordinate in the same virtual predefined map used for the simulation sets. The equivalent solver for motion planning employed for the simulation sets ([32–34,42,43]) were used for the real mobile manipulator. During the experimental sets, it was observed that the IMU attached to the mobile platform measured vibrations when the mobile platform moved. These disturbed accelerations from the sensor caused a prejudicial effect on the calculations of the dynamic tilt stability, so that simulation and real robot could be hardly compared. For this purpose, a recursive filter was implemented, thus reducing significantly reducing the noise detected by the IMU.

**Table 3.** Comparative scenarios in order to verify the effectiveness of the active stabilization by means of simulations.

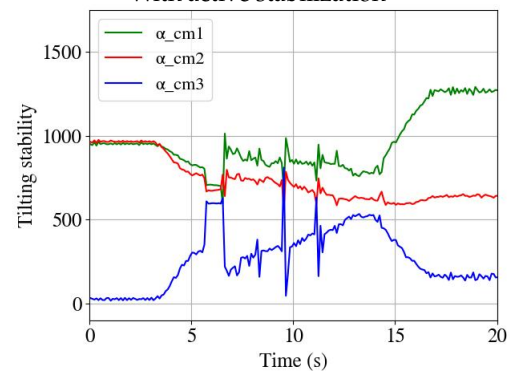
**Set 1**

Initial dynamic tilting stability  $\alpha = 23$



The first instability occurred at the time, the robot started to move (at 4 s). The robot tilted slightly over the front wheels. After a brief strong acceleration at 9 s, which was evident from the sudden increase in  $\alpha_{cm-3}$ , a deceleration occurred. Consequently, the mobile manipulator completely tipped over from 12 s onwards.

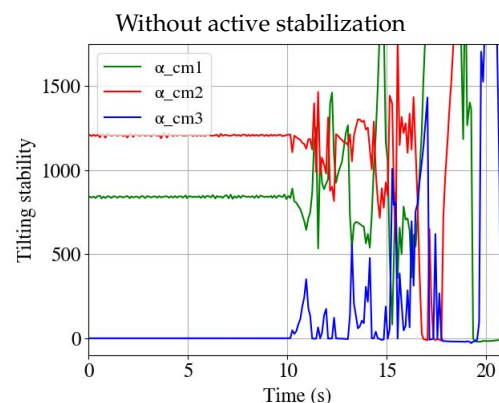
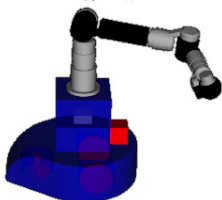
**With active stabilization**



A repositioning was performed before the mobile platform started to move, thus the decisive dynamic tilting stability increased to about  $\alpha_{cm-3} = 280$ . As long as the mobile platform was in motion, several repositioning courses were performed at 7 s, 8 s, 10 s, and 12 s, since the dynamic tilting stability went below the threshold  $\alpha_{cm-critical} = 250$ .

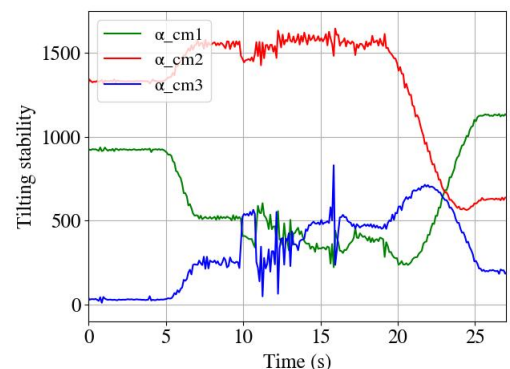
**Set 2**

Initial dynamic tilting stability  $\alpha = 17$



The acceleration process caused the mobile manipulator to tilt over the front edge several times between 10 s and 15 s. From about 15 s, the mobile manipulator tipped completely over the front edge.

**With active stabilization**

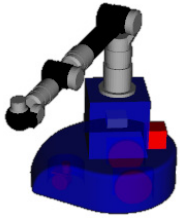


First, before the mobile platform moved forwards, a reposition process took place. The value of the decisive tilting stability  $\alpha_{cm-3}$  increased to just over 250. The mobile platform then started to move. A further stabilization took place at 12 s, increasing  $\alpha_{cm-3}$  by about 500. On the other hand, the  $\alpha_{cm-1}$  fell below the limit value at 16 s. This resulted in a new repositioning for the robot arm.

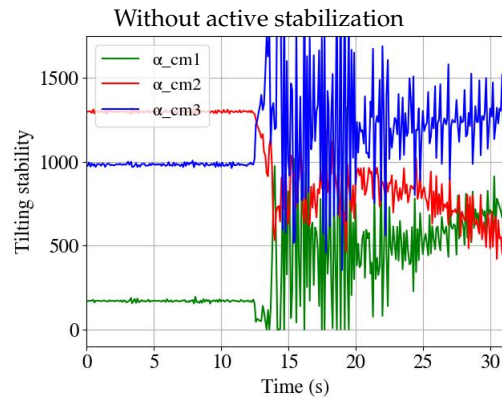
Table 3. Cont.

Set 3

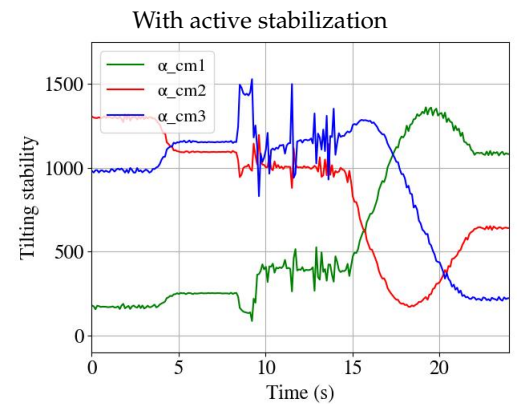
Initial dynamic tilting stability



The critical tipping axis was one of the two lateral tipping axes,  $\alpha_{cm-1} = 157$ , representing the critical tilt stability.



At about 13 s, the entire system tipped over the tilting edge. The bumpers of the mobile platform dragged across the floor, causing a strong noise from 13 s onward.



A repositioning occurred at about 4 s. The value for  $\alpha_{cm-1}$  increased thus to about 250. The acceleration process at about 7 s caused a destabilizing effect against the lateral tilting edge. A new reposition was initiated, increasing the value for  $\alpha_{cm-1}$  to about 400.

Analogous to the above simulations, the sets included in Table 4 differ from each other in terms of the initial position and orientation of the robot manipulator.

For all sets with active stabilization, the mobile manipulator reached the target position without any risk of instability. During sets 1 and 2, even a tip-over of the system was prevented. During the active stabilizations, the repositioning of the robot arm took place several times along the trajectory of the mobile platform to increase the stability of the entire system.

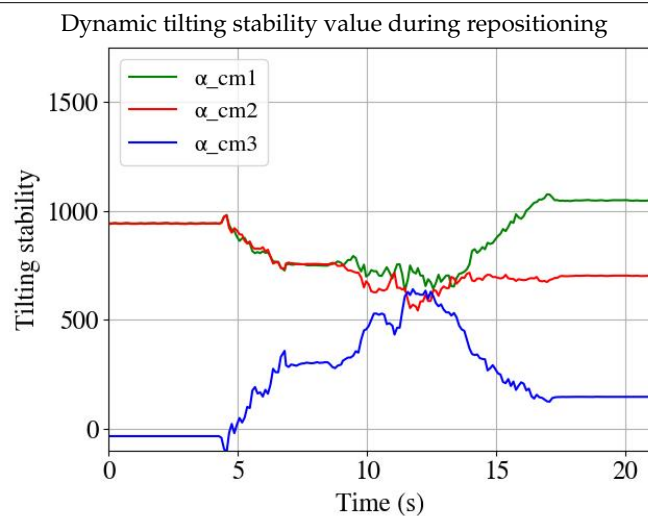
Table 4. Comparative scenarios in order to verify the effectiveness of the active stabilization using the real mobile manipulator.

Set 1

Initial position and orientation of robot arm



$\alpha = -34$

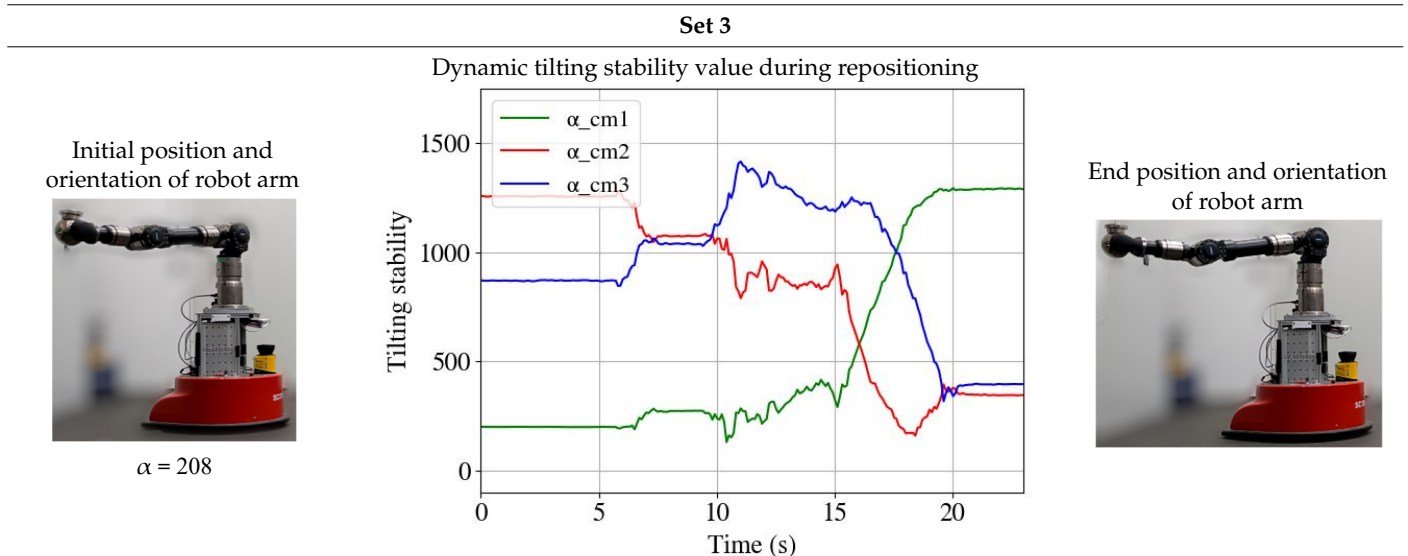
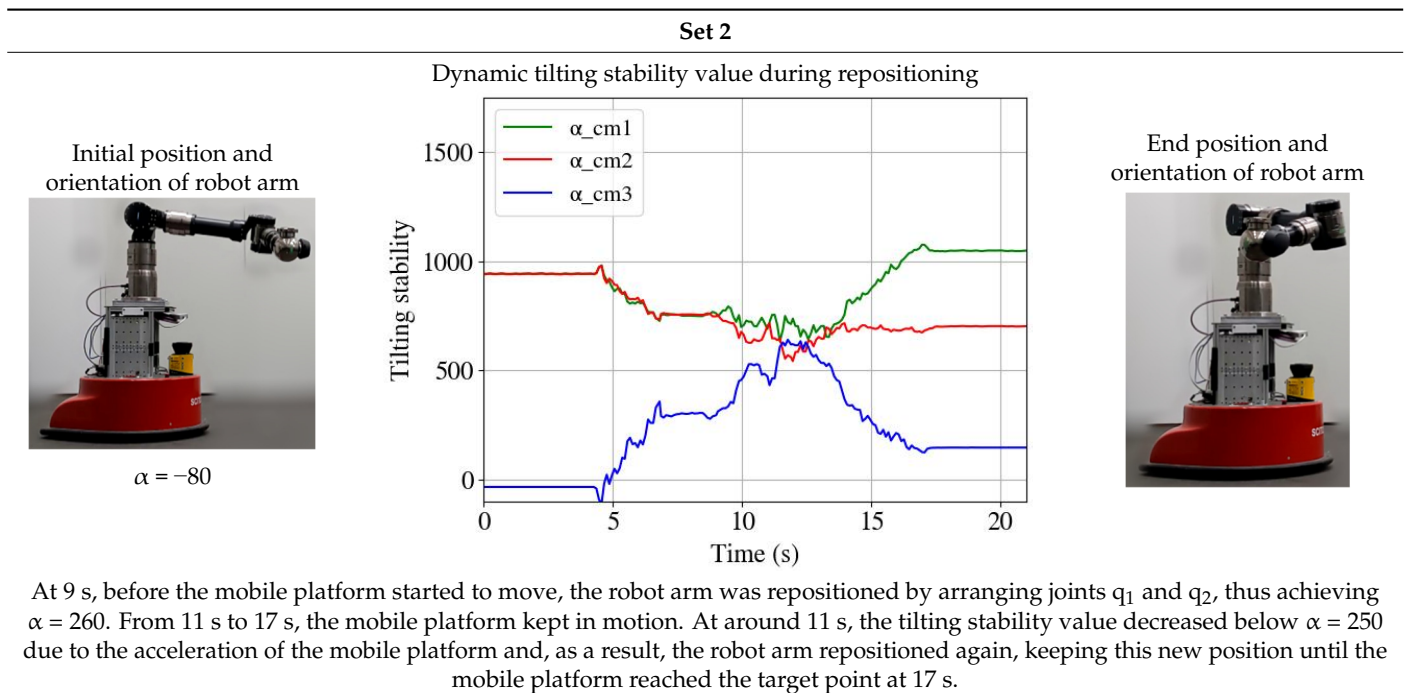


End position and orientation of robot arm



Between 5 s and 7 s, the robot arm was rearranged to achieve  $\alpha = 280$ , before the mobile platform started to move. During the trajectory, no additional repositioning was needed. The mobile manipulator achieved the target point at 16 s.

Table 4. Cont.



For all sets with active stabilization, the real mobile manipulator reached the target point without any instability risk and retained its original TPC orientation. Therefore, the proposed stabilization strategy satisfied these expectations.

Comparing the results obtained from the simulation sets and the sets employing the real mobile manipulator, it can be seen that, for all sets, an initial repositioning procedure had to be performed before the mobile platform started to move, since the mobile manipulator stability values were below the predefined threshold of the critical stability value.

In Set 1, additional repositioning procedures were carried out for the simulations, while in the real mobile manipulator no further repositioning was required, since the stability values remained above the threshold for the critical stability value. During Set 2, besides the initial repositioning, a second repositioning procedure was performed for both, simulation and real mobile manipulator. However, the simulation set demanded a third

repositioning during the braking process of the mobile platform, while the real robot did not. During Set 3, in addition to the initial repositioning, a second repositioning procedure was performed with respect to the lateral edge of the mobile platform in both, simulation and real mobile manipulator, generated during the acceleration of the mobile platform.

The tendency indicates that during the simulation sets, an extra repositioning procedure was required during the traveling or the braking process of the mobile platform. This discrepancy may be related to the acceleration variances of the mobile platform during the experimental sets with the real mobile manipulator. The theoretical acceleration and braking processes of the mobile platform, employed during the simulations, could not be exactly replicated during the real experimental sets, since the motion control algorithms of the manufacturer were not altered.

The observed differences in the initial value of the stability value between the simulations and experiments employing the real mobile manipulator may be attributed to the simplification procedure conducted during the modeling of the mobile manipulator for the simulations, already explained in previous sections.

## 5. Discussion

Small-footprint mobile manipulators offer not only high flexibility owing to their compact design, but also agility and maneuverability. As mentioned in the literature review, this type of systems tend to tip over because of their high accelerating and braking capability, as well as their high center of gravity.

The aim of this study was to implement a workspace optimization and an active stabilization by repositioning the robot arm. The implemented active stabilization strategy covers the following features:

- After receiving a target coordinate, the mobile manipulator independently determines whether the target is located within the predefined reachable and stable workspace or outside it. If the target point is located outside the optimized volume, the mobile platform approaches it.
- When instabilities are detected by the MHS before the mobile platform moves, the robot arm repositions itself to a non-critical position.
- If an instability is caused by accelerations or decelerations, which means that the stability value falls below the predefined threshold value, the robot arm repositions itself without aborting any current tasks. During this repositioning process, the initial orientation of the TPC is maintained.
- The robot arm approaches the target point only after the mobile platform reaches the target location. Thus, by extending the robot arm, it can be guaranteed that the TPC is situated within the stability-optimized workspace.

Perhaps the most significant added value with respect to previous studies is that the proposed countermeasures are carried out in real time without manipulating the motion planning of the robots, thus reducing the complexity of the implementation and increasing its flexibility as a universal solution.

The results from the evaluation sets of the stabilization strategy using the real mobile manipulator indicated that the proposed approach was able to avoid many tip-overs in the robotic system under different circumstances. In general, the presented active stabilization strategy can be adapted for all types of compact lightweight mobile manipulators, regardless of the model, size, and manufacturer. The only restriction imposed for its use is that both the mobile platform and the manipulator must be operated with ROS.

Future research is required to explore the impact of employing other testing models, not only for the manipulator and mobile platform but also for different types of path planning and motion control algorithms. In any case, there is a need for testing the approach in a wide variety of environments to reinforce the proposed algorithm.

## 6. Conclusions

This study aimed to evaluate the effectiveness of an active stabilization algorithm for compact and lightweight mobile manipulators. First, we calculated the dynamic stability value  $\alpha$  using the Moment Height Stability measure (MHS), which provided information about the system's stability state  $\alpha$ . This value represents the effects of the robot arm and mobile platform movements, COGs position changes, and gravity vector changes. The larger the value of  $\alpha$ , the more stable the system is against tip-over regarding the corresponding tilting edge for which the value was calculated. Therefore, the smaller the value of  $\alpha$ , the greater the risk of tipping over. Employing the MHS, we considered not only the direction of the overall acceleration but also its overall effect on the system, since the mobile manipulator could appear to be stable, despite the fact that the resulting inertial forces could cause the system to tip over. We considered the tip-over edge as a very important indicator for the implementation of suitable prevention actions. A countermeasure can also contribute to system tip-over if it is applied to an incorrect tilting edge.

Once the MHS indicates the stability state of the entire system, we optimized the theoretical workspace of the manipulator based on this stability value ( $\alpha$ ) to constrain the area in which the manipulator is able to operate without any risk of instability, as long as the mobile platform is not in motion. We proved the effectiveness of this simple optimization method by employing the real mobile manipulator. In addition, we developed a tip-over avoidance algorithm by repositioning the joints of the manipulator when the mobile manipulator is in motion. The repositioning algorithm determines a safe configuration for the manipulator (TCP) based on a gradient method. The method was configured so that the manipulator TCP was kept as close as possible to its original position. We validated its effectiveness by observing that the real mobile manipulator achieved its target position without presenting any instability.

The practically constant orientation of the TCP during the repositioning process as well as the avoidance of the risk of self-collision were significantly relevant for the active stabilization strategy. The deviations between the calculated tilting stability value and the value at which the system is truly tipping over were compensated by repetitive repositioning processes.

Although a particular mobile manipulator was employed for development and validation purposes, this study establishes a simple approach that can be applied to other models by simply computing their corresponding critical stability value and adapting the robot characteristics to the presented algorithms.

Future research is recommended to explore the impact of employing other testing models of mobile manipulators and their different types of path planning and motion control algorithms. In any case, the outcomes point to the need for testing in a wide variety of environments to reinforce the developed algorithms.

## 7. Contributions

The most significant contribution of our work with respect to previous studies is that the presented countermeasures are carried out without manipulating the motion planning of the robots, thus reducing the complexity in its implementation, and increasing its flexibility as a universal solution. Additionally, only the robot manipulator was employed to achieve the stabilization without having to abort any of the mobile platform motion planning tasks.

The repositioning procedure has been developed such that the robot manipulator TCP is kept as close as possible to its original position, avoiding mishandle of workpieces. A further distinction of the presented method is that the dynamic tilting stability value of the mobile manipulator was employed not only during the tip-over detection, but also for the tip-over avoidance, being this value the reference for the gradient potential function used for the calculation of the new joint configuration needed for the repositioning of the robot manipulator's links.

Despite the fact that the stabilization strategy was developed and validated on a particular real mobile manipulator, the method can be employed as a universal solution just by adjusting the corresponding robot parameters into the provided algorithms.

**Supplementary Materials:** The following supporting information can be downloaded at: <https://www.mdpi.com/article/10.3390/machines11010044/s1>. References [44–48] are cited in the Supplementary Materials.

**Author Contributions:** Conceptualization, A.T.F. and M.K.; methodology, A.T.F. and T.B.; software, F.K. and T.B.; validation, F.K. and T.B.; formal analysis, A.T.F. and F.K.; investigation, F.K.; resources, M.K.; writing—original draft preparation, A.T.F.; writing—review and editing, M.K. and M.J.P.; visualization, A.T.F. and F.K.; supervision, M.K. and M.J.P. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** The data presented in this study are available as Supplementary Material.

**Acknowledgments:** The authors would like to thank Thomas Bertram and Maximilian Bryg for their technical support to this research. We are also grateful to the anonymous reviewers of this article for their very constructive contributions.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Rey, D.A.; Papadopoulos, E.G. On-line automatic tipover prevention for mobile manipulators. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Grenoble, France, 11 September 1997. [CrossRef]
2. Tahboub, K.A. Robust control of mobile manipulators. *J. Robot. Syst.* **1996**, *13*, 699–708. [CrossRef]
3. Ding, X.; Liu, Y.; Hou, J.; Ma, Q. Online Dynamic Tip-Over Avoidance for a Wheeled Mobile Manipulator with an Improved Tip-Over Moment Stability Criterion. *IEEE Access* **2019**, *7*, 67632–67645. [CrossRef]
4. Böge, A.; Böge, W. *Technische Mechanik: Statik—Reibung—Dynamik—Festigkeitslehre—Fluidmechanik*; Springer Vieweg: Wiesbaden, Germany, 2019; Volume 33, ISBN 978-3-658-25724-8.
5. Ghasempour, A.; Sepehri, N. Measure of machine stability for moving base manipulators. In Proceedings of the IEEE International Conference on Robotics and Automation, Nagoya, Japan, 21–27 May 1995. [CrossRef]
6. Li, Y. Dynamic stability analysis and control for the mobile manipulator. In Proceedings of the IEEE CCECE2002 Canadian Conference on Electrical and Computer Engineering, Winnipeg, MB, Canada, 12–15 May 2002. [CrossRef]
7. Papadopoulos, E.G.; Rey, D.A. New measure of tipover stability margin for mobile manipulators. In Proceedings of the IEEE International Conference on Robotics and Automation, Minneapolis, MN, USA, 22–28 April 1996. [CrossRef]
8. Papadopoulos, E.; Rey, D.A. Force-angle measure of tipover stability margin for mobile manipulators. *Veh. Syst. Dyn.* **2000**, *33*, 29–48. [CrossRef] [PubMed]
9. Vukobratović, M.; Borovac, B. Zero-Moment Point—Thirty Five Years of its Life. *Int. J. Humanoid Robot.* **2004**, *1*, 157–173. [CrossRef]
10. Sugano, S.; Huang, Q.; Kato, I. Stability criteria in controlling mobile robotic systems. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '93), Yokohama, Japan, 26–30 July 1993. [CrossRef]
11. Huang, Q.; Sugano, S.; Kato, I. Stability control for a mobile manipulator using a potential method. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94), Munich, Germany, 12–16 September 1994. [CrossRef]
12. Moosavian, S.A.A.; Alipour, K. On the dynamic tip-over stability of wheeled mobile manipulators. *Int. J. Robot. Autom.* **2007**, *22*, 322. [CrossRef]
13. Moosavian, S.A.A.; Alipour, K. Stability evaluation of mobile robotic systems using moment-height measure. In Proceedings of the 2006 IEEE Conference on Robotics, Automation and Mechatronics, Bangkok, Thailand, 1–3 June 2006. [CrossRef]
14. Moosavian, S.A.A.; Alipour, K. Moment-Height tip-over measure for stability analysis of mobile robotic systems. In Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China, 9–15 October 2006. [CrossRef]
15. Roan, P.R.; Burmeister, A.; Rahimi, A.; Holz, K.; Hooper, D. Real-world validation of three tipover algorithms for mobile robots. In Proceedings of the IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 3–7 May 2010. [CrossRef]
16. Huang, Q.; Sugano, S. Manipulator motion planning for stabilizing a mobile-manipulator. In Proceedings of the 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots, Pittsburgh, PA, USA, 5–9 August 1995. [CrossRef]
17. Alipour, K.; Hasanpour, A.; Daemy, P. Comparing two online tip-over avoidance algorithms for mobile manipulators. In Proceedings of the 2014 2nd RSI/ISM International Conference on Robotics and Mechatronics, ICRoM 2014, Tehran, Iran, 15–17 October 2014. [CrossRef]



18. He, L. Tip-over avoidance algorithm for modular mobile manipulator. In Proceedings of the 2012 First International Conference on Innovative Engineering Systems, ICIES 2012, Alexandria, Egypt, 7–9 December 2012. [CrossRef]
19. Hatano, M.; Obara, H. Stability evaluation for mobile manipulators using criteria based on reaction. In Proceedings of the SICE 2003 Annual Conference (IEEE Cat. No.03TH8734), Fukui, Japan, 4–6 August 2003.
20. Furuno, S.; Yamamoto, M.; Mohri, A. Trajectory planning of mobile manipulator with stability considerations. In Proceedings of the IEEE International Conference on Robotics and Automation (Cat. No.03CH37422), Taipei, Taiwan, 14–19 September 2003. [CrossRef]
21. Kim, J.; Chung, W.K.; Youm, Y.; Lee, B.H. Real-time ZMP compensation method using null motion for mobile manipulators. In Proceedings of the 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292), Washington, DC, USA, 11–15 May 2002. [CrossRef]
22. Li, Y.; Liu, Y. Fuzzy logic self-motion planning and robust adaptive control for tip-over avoidance of redundant mobile modular manipulators. In Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM, Monterey, CA, USA, 24–28 July 2005. [CrossRef]
23. Li, Y.; Liu, Y. Real-time tip-over prevention and path following control for redundant nonholonomic mobile modular manipulators via fuzzy and neural-fuzzy approaches. *J. Dyn. Syst. Meas. Control. Trans. ASME* **2006**, *128*, 753–764. [CrossRef]
24. Ettorre, C.D.; Stilli, A.; Dwyer, G.; Neves, J.; Tran, M.; Stoyanov, D. Semi-Autonomous Interventional Manipulation using Pneumatically Attachable Flexible Rails. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; IEEE: New York, NY, USA; pp. 1347–1354. [CrossRef]
25. Ratliff, N.; Zucker, M.; Bagnell, J.; Srinivasa, S. CHOMP: Gradient optimization techniques for efficient motion planning. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; IEEE: New York, NY, USA; pp. 489–494. [CrossRef]
26. Campana, M.; Lamiroux, F.; Florent, L.; Laumond, J.-P. A gradient-based path optimization method for motion planning. *Adv. Robot.* **2016**, *30*, 1126–1144. [CrossRef]
27. Khatib, O. Real-time obstacle avoidance for manipulators and mobile robots. In Proceedings of the 1985 IEEE International Conference on Robotics and Automation, St. Louis, MO, USA, 25–28 March 1985; pp. 500–505. [CrossRef]
28. Raj, J.; Raghuwairi, K.; Sharma, B.; Vanualailai, J. Motion control of a flock of 1—Trailer robots with swarm avoidance. *Robotica* **2021**, *39*, 1926–1951. [CrossRef]
29. Raj, J.; Raghuwairi, K.S.; Vanualailai, J. Novel Lyapunov-Based Autonomous Controllers for Quadrotors. *IEEE Access* **2020**, *8*, 47393–47406. [CrossRef]
30. MIRA: Concepts. Available online: <https://www.mira-project.org/MIRA-doc/ConceptsPage.html> (accessed on 16 October 2021).
31. STRANDS · GitHub. Available online: <https://github.com/strands-project> (accessed on 17 October 2021).
32. Schunk\_Robots—ROS Wiki. Open Robotics. Available online: [http://wiki.ros.org/schunk\\_robots](http://wiki.ros.org/schunk_robots) (accessed on 23 October 2021).
33. Fraunhofer IPA, GitHub—ipa320/schunk\_modular\_robotics. Available online: [https://github.com/ipa320/schunk\\_modular\\_robotics](https://github.com/ipa320/schunk_modular_robotics) (accessed on 16 October 2021).
34. GitHub—ipa320/schunk\_robots. Available online: [https://github.com/ipa320/schunk\\_robots](https://github.com/ipa320/schunk_robots) (accessed on 23 October 2021).
35. Becker, M.; Domnik, N. *3D Volume Modeling of a Mobile Robot as the Basis of a Multi-Body Simulation*; Hochschule Karlsruhe University of Applied Science: Karlsruhe, Germany, 2018. (In German)
36. Spiers, A.; Khan, S.G.; Herrmann, G. *Biologically Inspired Control of Humanoid Robot Arms: Robust and Adaptive Approaches*; Springer International Publishing: Cham, Switzerland, 2016; ISBN 9783319301600.
37. Quigley, M.; Gerkey, B.; Smart, W.D. *Programming Robots with ROS*; O'Reilly Vlg. GmbH & Co.: Seattle, WA, USA, 2015; ISBN 9781449323899.
38. Möllmann, S. *Integration of an Inertial Measurement Unit into the Autonomous Vehicle Platform CampusBot*; Hamburg University of Applied Sciences: Karlsruhe, Germany, 2014. (In German)
39. Featherstone, R. *Rigid Body Dynamics Algorithms*, 1st ed.; Springer: New York, NY, USA, 2008; ISBN 978-0-387-74314-1.
40. KDL Wiki | The Orocos Project. Available online: <https://www.orocos.org/kdl.html> (accessed on 17 October 2021).
41. Smits, R. orocos\_kdl: KDL::ChainIdSolver\_RNE Class Reference. Available online: [http://docs.ros.org/en/kinetic/api/orocos\\_kdl/html/classKDL\\_1\\_1ChainIdSolver\\_\\_RNE.html](http://docs.ros.org/en/kinetic/api/orocos_kdl/html/classKDL_1_1ChainIdSolver__RNE.html) (accessed on 17 October 2021).
42. Company, S.R. Planners Benchmarking Documentation Shadow Robot Company. 2020. Available online: <https://buildmedia.readthedocs.org/media/pdf/planners-benchmarking/latest/planners-benchmarking.pdf> (accessed on 30 November 2022).
43. Moll, M.; Sucan, I.A.; Kavraki, L.E. Benchmarking Motion Planning Algorithms: An Extensible Infrastructure for Analysis and Visualization. *IEEE Robot. Autom. Mag.* **2015**, *22*, 96–102. [CrossRef]
44. GitHub—ros-controls/ros\_controllers: Generic Robotic Controllers to Accompany ros\_control. Available online: [https://github.com/ros-controls/ros\\_controllers](https://github.com/ros-controls/ros_controllers) (accessed on 30 November 2022).
45. GitHub—cburbridge/scitos\_common. Available online: [https://github.com/cburbridge/scitos\\_common](https://github.com/cburbridge/scitos_common) (accessed on 30 November 2022).
46. GitHub—Strands-Project/Scitos\_drivers: Scitos G5 Drivers that Interface ROS to MIRA. Available online: [https://github.com/strands-project/scitos\\_drivers](https://github.com/strands-project/scitos_drivers) (accessed on 21 January 2022).

47. [1604.04384] The STRANDS Project: Long-Term Autonomy in Everyday Environments. Available online: <https://arxiv.org/abs/1604.04384> (accessed on 30 November 2022).
48. Navigation/Tutorials/SendingSimpleGoals. Available online: [http://library.isr.ist.utl.pt/docs/roswiki/navigation\(2f\)Tutorials\(2f\)SendingSimpleGoals.html](http://library.isr.ist.utl.pt/docs/roswiki/navigation(2f)Tutorials(2f)SendingSimpleGoals.html) (accessed on 30 November 2022).

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.